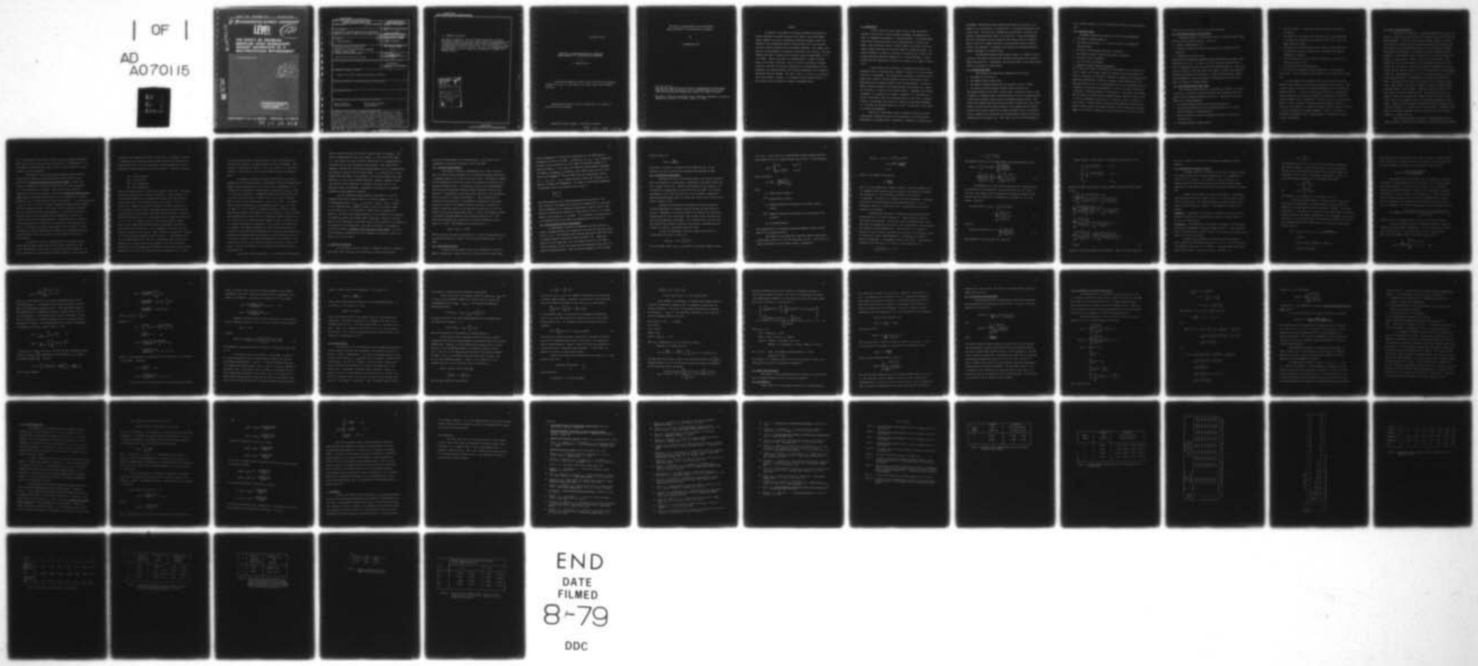
AD-A070 115 ILLINOIS UNIV AT URBANA-CHAMPAIGN COORDINATED SCIENCE LAB F/G 9/2
THE EFFECT OF PROGRAM BEHAVIOR UPON INTERLEAVED MEMORY BANDWIDT--ETC(U)
SEP 78 B R RAU DAAB07-72-C-0259
UNCLASSIFIED R-826 NL

| OF |

AD
A070115



END
DATE
FILMED
8-79
DDC

1.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

REPORT R-826 SEPTEMBER, 1978 UILU-ENG 78-2219

COORDINATED SCIENCE LABORATORY

LEVEL II

12

AD A070115

THE EFFECT OF PROGRAM BEHAVIOR UPON INTERLEAVED MEMORY BANDWIDTH IN A MULTIPROCESSOR ENVIRONMENT

B. RAMAKRISHNA RAU

UNIVERSITY OF ILLINOIS - URBANA, ILLINOIS

79 06 18 012

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER (9)
4. TITLE (and Subtitle) (6) THE EFFECT OF PROGRAM BEHAVIOR UPON INTERLEAVED MEMORY BANDWIDTH IN A MULTIPROCESSOR ENVIRONMENT		5. TYPE OF REPORT & PERIOD COVERED Technical Report
		6. PERFORMING ORG. REPORT NUMBER (14) R-826, UILU-ENG-78-2219
7. AUTHOR(s) (10) B. Ramakrishna Rau		8. CONTRACT OR GRANT NUMBER(s) (15) DAAB-07-72-C-0259
9. PERFORMING ORGANIZATION NAME AND ADDRESS Coordinated Science Laboratory University of Illinois at Urbana-Champaign Urbana, Illinois 61801		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS (12) 59p.
11. CONTROLLING OFFICE NAME AND ADDRESS Joint Services Electronics Program		12. REPORT DATE (11) September, 1978
		13. NUMBER OF PAGES 42
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Memory Bandwidth; Interleaved Memory; Multiprocessor Systems; Program Behavior		

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

An analysis is performed of the memory bandwidth that results in a system consisting of multiple processors and a shared, interleaved memory. Special emphasis is placed on studying and modeling the memory referencing behavior of the programs that execute on the processor and in ascertaining the effect that this behavior has on the memory bandwidth. Some commonly used models of program behavior are shown to be unrealistic and, instead, a new, more sophisticated model is proposed and validated by means of measurements on four trace tapes. Based on this model of program behavior, a simple structural model of

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

097 700 JOB

20. ABSTRACT (continued)

the system is analyzed. This is effected by extending a technique previously developed for use with a simpler model of program behavior and by advancing a new technique based on the diffusion approximation and probability generating functional methods. The latter analysis is shown, via simulations, to be very accurate. Finally, it is demonstrated that program behavior, in this context, may be captured by a simple, two-parameter model.

UILU-ENG 78-2219

THE EFFECT OF PROGRAM BEHAVIOR UPON INTERLEAVED MEMORY BANDWIDTH IN A MULTIPROCESSOR ENVIRONMENT

by

B. Ramakrishna Rau

This work was supported in part by the Joint Services Electronics Program (U.S. Army, U.S. Navy and U.S. Air Force) under Contract DAAB-07-72-C-0259.

79 06 18 012

The effect of program behavior upon interleaved memory bandwidth in a multiprocessor environment[†]

by

B. Ramakrishna Rau[*]

[†]This work was supported in part by the U.S. Energy Research and Development Administration under Contract No. EY-76-S-03-0236-PA39 and in part by the Joint Services Electronic Program under Contract No. DAAB-07-72-C-0259.

[*]The author is with the Coordinated Science Laboratory, Department of Electrical Engineering, University of Illinois, Urbana, IL 61801.

ABSTRACT

An analysis is performed of the memory bandwidth that results in a system consisting of multiple processors and a shared, interleaved memory. Special emphasis is placed on studying and modeling the memory referencing behavior of the programs that execute on the processor and in ascertaining the effect that this behavior has on the memory bandwidth. Some commonly used models of program behavior are shown to be unrealistic and, instead, a new, more sophisticated model is proposed and validated by means of measurements on four trace tapes. Based on this model of program behavior, a simple structural model of the system is analyzed. This is effected by extending a technique previously developed for use with a simpler model of program behavior and by advancing a new technique based on the diffusion approximation and probability generating functional methods. The latter analysis is shown, via simulations, to be very accurate. Finally, it is demonstrated that program behavior, in this context, may be captured by a simple, two-parameter model.

1. Introduction

In view of the fact that a large fraction of the instructions executed in any processor require operands from memory, the memory access time is a fundamental determinant of overall performance. The average memory access time is directly affected by the technology used to construct the memory. When performance requires a shorter memory access time, one may use either a faster technology for the entire memory or a memory hierarchy. The latter alternative is, generally, the more cost-effective and is widely used in many medium and high speed computers [1-5]. The effectiveness of such a strategy has been demonstrated in theory and in practice by a number of workers [6-11].

A second factor that contributes to the memory access time is the degradation that is experienced when a number of requests vie with one another for the use of the memory. The solution that is employed is to partition the memory into a number of distinct physical resources called modules, each of which may service a request in parallel with the other modules. The sets of memory locations contained in the various modules are disjoint and a request can, therefore, be served only by a specific module. Depending on the pattern of requests, it is possible for a number of requests to be conflicting for a subset of the modules, while the remaining modules lie idle with a consequent increase in the average access time and a reduction in the memory bandwidth, i.e., the mean rate at which requests are served. The objective of this paper is to study this phenomenon.

The use of a cache memory reduces the memory interference problem at main memory substantially since the block-fetches to the cache exercise main memory in an optimal fashion - sequentially. However, in a multiprocessor environment, interference between requests takes place at the cache if it is common to all the processors. An alternative is to provide each processor with its own cache. While this does eliminate the interference,it introduces the problem of updating multiple copies of the same item. Every store operation must be cross-checked with every cache and since stores account for approximately a fifth of all memory references, the interference problem is merely reduced, not eliminated. Consequently, the analysis of interleaved memories remains a worthwhile and relevant study.

The bandwidth obtained from the memory depends on a number of factors relating to the memory sub-system, the processor sub-system and the processor-memory interconnection. Consequently, the structural model of the system should, ideally, incorporate all these factors.

1.1. The processor model

The processor sub-system may be parameterized as follows:

1. the number of processors, N,
2. the maximum request issuance rate,
3. the number of pending requests that the processor is able to buffer,
4. the interval between successive requests from a processor, etc.

The tendency appears to have been to study either a uniprocessor which is able to generate an unlimited number of pending requests [12-19] or a multiprocessor in which each processor can have only one outstanding request [20-27]. Most studies have assumed that each processor generates a request every cycle unless blocked but there have been some exceptions; Flores assumes that the request arrivals form a Poisson process [12], Baskett and Smith allow for a processor to wait a geometrically distributed time between having a request serviced and generating the next request [24]. This "think" time has also been assumed to be of constant length [21, 23] or distributed arbitrarily with a given mean [25].

1.2. The memory model

The memory sub-system is described by the following parameters:

1. the number of memory modules, M,
2. the width of each module, i.e., the unit of access of information from the module,
3. the type of interleaving - high-order or low-order,
4. the extent and type of buffers for queueing of requests,
5. the access time of the memory,
6. the cycle time of the memory,
7. whether the modules operate synchronously.

By and large, the width of the module has been assumed to be the same as the unit of request, but Coffman, Burnett and Snowdon study the case where the width of the module is a multiple of the unit of request by the processor [15]. High-order interleaving, in which all locations, whose addresses have the same high-order bits, go into the same module, has certain advantages from the viewpoint of reliability and availability. However, low-order interleaving is employed if enhanced bandwidth is the objective. Many models have assumed that the memory sub-system cannot accomodate requests awaiting service [13,14, 17,19] and, accordingly, the processor is blocked as soon as a request is to a busy module. Flores assumes an infinitely large queueing buffer per module whereas Coffman, Burnett and Snowdon consider a finite sized conflict buffer common to the modules. Although most work has assumed that the cycle time and access time are identical there have been some exceptions.

1.3. The processor-memory interconnection

Although many types of interconnections may be conceived of, just a handful have received serious attention. These are:

1. a shared, time-multiplexed bus,
2. a crossbar connection such that every processor is connected to every module and
3. a combination of the above two.

The majority of uniprocessor bandwidth studies have implicitly assumed a time-multiplexed bus [12-19]. Of these, all but [19] have assumed that the bus cycle is negligible and that all requests are made simultaneously. Most multiprocessor studies, on the other hand, have assumed a crossbar interconnection between the processors and the memories. Briggs and Davidson [28], however, have investigated a system consisting of ℓ time-multiplexed busses each of which is connected to m memory modules.

1.4. The structural model under study

The structural model which will be analyzed in this paper is a simplification of one presented by Skinner and Asher [20] and which has been frequently used by subsequent workers [21-24,26,27]. Specifically, the model makes the following assumptions:

1. the system contains N statistically identical processors,
2. each processor may have only one outstanding request and is blocked while the request is awaiting service,
3. when a processor is serviced it immediately generates a request the very next cycle,
4. the system contains M memory modules,

5. the module width is 1 double word (8 bytes) which is also the unit of request,
6. the modules are interleaved on the low-order bits of the address that is obtained after discarding the lowest-order three bits (since the module width is 8 bytes),
7. the memory has no queueing buffer space; however, the processors may be conceptually viewed as being queued on the module that they are attempting to reference,
8. the access time and the cycle time of the modules are identical and serve as the unit of time,
9. the modules operate synchronously and
10. the processors and memory are interconnected by a crossbar switch.

Such a model is fairly successful as an abstraction of a system such as C.mmp [29].

An issue that has been neglected in the above discussion of system modeling is the behavior of the references generated by each processor. This reference pattern can affect the bandwidth substantially and is discussed at length in Section 2. Section 3 reviews various analytical techniques that have been employed in the past and attempts to extend them to the model of reference behavior that is assumed here. Section 4 develops an exact analysis which, by itself, is unable to provide the desired results. However, in Section 5, this analysis, in conjunction with the diffusion analysis of Section 3, yields the expression for memory bandwidth. Finally, in Section 6, a simplified two-parameter model of program behavior is advanced and shown to be quite satisfactory.

2. A Model of Program Behavior

A perusal of the literature on interleaved memories reveals very few attempts to measure and study the effects of program behavior upon interleaved memory bandwidth. In the area of uniprocessor systems, Burnett and Coffman [14] modeled the reference string by separately modeling the instruction and data references and assuming that alternate cycles are dedicated to making instruction or data requests. The instruction stream is described by the parameter λ, the probability that an instruction is a successful branch (presumably to a target which is equally likely to lie in any module). With probability $(1-\lambda)$, the next instruction is the sequential successor of the current one. The model for the data stream employs a parameter α, the probability that the next data request will be to the sequentially following module (modulo M). With probability $(1-\alpha)/(M-1)$ the next data request is to any one of the remaining modules. Although unable to analytically evaluate this model in the general case, Burnett and Coffman did so via extensive simulations. Unfortunately, they did not validate this model of program behavior with measurements on real programs. Terman has tested the validity of this model using trace-driven simulations [18]. The technique he used was to measure λ and α and use Burnett and Coffman's results to predict the bandwidth. This he then compared with direct simulation measurements of bandwidth. Terman's results indicate that whereas the model is adequate for the instruction stream, it is ineffective for the data stream. Furthermore, the model is not directly applicable when the instruction and data streams are merged as is normally the case.

The Least-Recently-Used Stack Model of program behavior has been used as a model of memory referencing behavior in the uniprocessor environment

[19]. The advantage of this model is that it is able to capture the property which leads to the clustering in time of references to the same module. It is also amenable to analysis and has been shown to be quite accurate in its predictions of observed bandwidth.

The tendency in most multiprocessor bandwidth studies has been to assume that the sequence of requests generated by each processor can be modeled by the Random Independent Reference Model, (RIRM), in which every request is independent of every other request and has an equal probability, 1/M of being directed to any module. There have been two exceptions; Hoogendoorn [25] models the reference string by the Independent Reference Model, (IRM), whereby each request from processor i has an independent and constant probability, p_{ij}, of being directed to module j where $\sum_j p_{ij} = 1$. p_{ij} need not be equal to p_{kj} for $i \neq k$. He is able to perform an approximate analysis which compares well with simulation results. Sethi and Deo [26] have employed a localized model of referencing in which with probability α, a processor references the same module that it last referenced and with probability $(1-\alpha)/(M-1)$ it references any one of the remaining modules. They propose this as a model of referencing behavior in a memory that is interleaved on the high-order bits. They do not validate their model by measurements nor are they able to derive an expression for memory bandwidth in the general case, although they do form conjectures based on the exact results for small numbers of processors or memory modules.

In developing a model of program behavior, one must take care to ensure that the model captures most of the relevant characteristics that affect bandwidth while at the same time retaining analytic tractability. The first step taken was to check whether the RIRM or IRM were in fact sufficient or whether a more sophisticated model was called for. Accordingly, a series of simulations were performed in which systems consisting of 4 processors and M memory modules ($M = 2,4,8,16$) were studied. Four trace tapes were used to generate the request sequences for the four processors. These trace tapes are:

043 - Fortran execution

049 - Sort execution

050 - Cobol compilation

051 - Cobol compilation

These trace tapes are part of a much larger library of tapes [30]. The memory modules were assumed to have a width of 8 bytes (i.e., the low-order 3 bits of each request address were discarded) and they were interleaved based on the low-order bits of the remaining portion of the address. Since it is possible for a number of processors to reference a module in the same cycle it is necessary to define a priority by which requests are entered into the queue associated with the module. The scheme used was one with a "rotating priority" such that in cycle j, the processors would be inspected in the order j, $j+1$, $j+2$, $j+3$ (modulo 4) to see if they had any request to make, in which case the request would be entered into the appropriate queue before proceeding to the next processor. As a consequence, no processor gets preferential treatment in the long run. Other than this, the simulation conformed to all the structural assumptions listed in Section 1. Each simulation was run for 10,000 memory cycles. Two sets of measurements were taken for each simulation run; firstly, the long run probability of referencing a module averaged over all four programs was estimated to allow for the testing of RIRM hypothesis and, secondly, the long run conditional probability q_{ij} of a processor referencing module j given that module i was last referenced by the processor was estimated, once again averaged over all four programs. These statistics are presented in Tables 1, 2 and 3 for M equal to 2, 4 and 8 respectively. (The statistics for $M = 16$ were gathered but are not presented here).

An examination of these three tables reveals that the long term probability of referencing a module is approximately 1/M and an IRM model with unequal reference probabilities is ruled out. Next, an examination of the estimated first-order Markov transition matrices shows that $q_{ij} \neq q_{kj}$ for all i,k. Hence, the hypothesis of independence between successive requests must be rejected and along with it the RIRM. Finally, the localized reference model of Sethi and Deo requires that $q_{ij} = q_{ik}$ for all j, $k \neq i$. The measurements indicate that whereas q_{ij} is approximately equal to q_{ik} for j and k not equal to i or $i+1$, there is a pronounced tendency to reference either the same module again or the sequential successor (modulo M). The localizing effect hypothesized by Sethi and Deo in a high-order interleaved memory is in fact present in the low-order interleaved memory too, but, in addition, there is a significant measure of sequentiality in the reference string generated by a program. The enhanced probability of referencing the same module is explained by operations of the read-modify-write type and by byte manipulating instruction (SS instructions) with separately access the individual bytes in a double word (8 bytes). The observed sequentiality may be explained by the sequentiality inherent in the instruction stream as well as by operations on arrays. That the data stream, too, has significant sequentiality has been demonstrated [31].

On the basis of these measurements it is clear that the RIRM and IRM must be rejected and that the localized reference model is inadequate. The model of program behavior used in this paper is a first-order Markov model, i.e., one in which the module reference probabilities for a program depend on the module that was last referenced by that program. Such a model may be expected to capture both localized referencing behavior as well as sequentiality and contains as special cases the IRM and RIRM. It is possible and, in fact, probable that this model is not adequate in the sense that higher order dependencies do, in fact, exists between references. However, subsequent results will show that such dependencies, to whatever extent they exist, do not materially affect the bandwidth. Furthermore, this model has the important advantage of being amenable to analysis.

One last observation which simplifies the analysis is that the estimated first-order transition matrix is circularly symmetric, i.e., q_{ij} is approximately equal to $q_{i+k, j+k}$ for all i, j and k. This, of course, is to be expected since a program is equally likely to be placed in memory at one location as it is to be placed in another which is displaced by k modulo M. The first-order transition matrix is fully described by M parameters, p_0 through p_{M-1} where p_k is defined equal to $\sum_i q_{i,i+k}/M$. Thus p_0 is the probability of re-referencing the same module and p_1 is the probability of referencing the sequentially following module. This model, described by the parameters $\{p_i\}$ will be termed the Symmetric First-Order Markov Model (SFMM). The values of the parameters averaged over all four programs are presented in Table 4 for values of M of 2,4,8, and 16.

3. Analytical techniques

In this section we review a number of techniques, mostly of an approximate nature, which have been used in analyzing the same structural model but with the RIRM assumption for program behavior. Our interest lies in attempting to extend these techniques to handle the SFMM.

3.1. Strecker's approximation

The essence of Strecker's approximation [21] is that a processor that is blocked at memory during one cycle discards that request and proceeds to generate requests as if it had in fact been served. The expression obtained for the bandwidth only takes into account the combinatorial interference and ignores the queueing effect. The equivalent analysis for the SFMM assumes that each processor generates requests without interference from the other processes. The probability of any given module being idle is the probability that no processor is currently referencing it. Since the generation of references by each processor is, by assumption, independent of every other processor's activity, the probability that no processor is referencing a module is simply the product of the individual probabilities for the N processors of not referencing the given module. In view of the symmetric nature of the model of program behavior, this individual probability is $(1-1/M)$. Hence the probability that no processor is currently referencing a given module is $(1-1/M)^N$. The bandwidth is accordingly given by

$$B(M,N) = M[1 - (1 - 1/M)^N]$$

which is exactly the same as for the RIRM case. Strecker's approximation does not permit the analysis to benefit from the increased sophistication of the SFMM.

3.2. Exact Markov analysis

An exact analysis similar to that used by Bhandarkar [23] for the RIRM is a possibility. However, the size of the state space is much larger since the aggregation of states that is permissible for the RIRM cannot be performed in the case of the SFMM. In particular, let $<n_0,\dots,n_{M-1}>$ represents the state of the system where n_i is the number of processors enqueued or in service at module i. In the case of the RIRM, any state obtained by permuting the numbers n_0 through n_{M-1} is equivalent to the original state and all M! such states may be aggregated. With the SFMM, only cyclic permutations, such as $<n_i \cdots\cdots n_{M-1}, n_0,\dots\dots n_{i-1}>$, produce equivalent states and, therefore, only groups of M states can be aggregated. The number of states before aggregation of equivalent states can be shown [32] to be equal to

$$\binom{N+M-1}{M-1}$$

and after aggregation it is reduced approximately by the factor M. The exact solution of the Markov chain requires the enumeration of all the states and the computation of the transition probabilities along the lines of [23]. This can be prohibitively expensive for large values of M or N or if a large number of (M,N) pairs are to be studied. It is preferable to develop analytical techniques which do not require the enumeration of the state space.

3.3. The exponential queueing approximation

An approximation introduced by Bhandarkar and Fuller [22] was to replace each memory module by an exponential server with the same mean service time of 1 cycle. The resulting closed quencing system falls within the class of queueing networks considered by Gordon and Newell [33]. Using their results it can be shown that, since the transition matrix for the SFMM is doubly stochastic and since all servers have the same mean service time, all states are equally probable. Then by using the arguments in [22] the bandwidth is found to be

$$B(M,N) = \frac{NM}{N+M-1}$$

This result, once again, is unchanged from the RIRM case and, so, this analysis, too, must be discarded as inadequate when studying the SFMM.

3.4. The diffusion approximation

The source of error in the previous analysis was that the structural model was compromised by substituting a exponentially distributed service time for a constant service time. However, this permitted an exact analysis. An alternative is to approximate the analysis but not the model. The diffusion approximation [34,35] does just this and is the only general technique for handling queueing networks that fall outside the class of the so-called "local balance" networks [36,37].

Kobayashi [38] and Reiser and Kobayashi [39] have developed the diffusion approximation for general queueing networks and have studied its accuracy. Bhandarkar and Fuller have applied this theory to the interleaved memory problem [22] and, surprisingly, they obtained exactly the same result as with the exponential server model. This is probably due to the fact that they neglected the boundary conditions in the analysis, by paying attention to which one obtains a considerably different result.

In [39], the approximate joint queue length distribution for a closed network of queues is shown to be

$$\hat{p}(n_1,n_2,\ldots,n_M) = \pi \prod_{i=1}^{M} \tilde{p}_i(n_i)$$

for all feasible states $(n_1,n_2,\ldots n_M)$ where n_i is the queue length at server i, $n_1 + n_2 + \ldots + n_M = N$, and π is a normalization constant chosen so that the total probability over all feasible states sums to unity. In this expression

$$\tilde{p}_i(n) = \begin{cases} 1 - \tilde{u}_i & \text{for } n = 0 \\ \tilde{u}_i (1 - \hat{\rho}_i) \hat{\rho}_i^{n-1} & \text{for } n \geq 1 \end{cases}$$

and $\hat{\rho}_i$ is given by

$$\hat{\rho}_i = \exp\left(-\frac{2(\mu_i - \lambda^{(i)})}{C_a^{(i)} \lambda^{(i)} + C_i \mu_i}\right)$$

where

μ_i = service rate of server i,

$\lambda^{(i)}$ = arrival rate to queue i,

C_i = squared coefficient of variation of the service time at server i,

$C_a^{(i)}$ = squared coefficient of variation of the inter-arrival time at queue i

$\tilde{u}_i$ = an unknown quantity.

Other expressions for calculating $\hat{\rho}_i$ have been suggested [40,41] and they generally yield greater accuracy.

The symmetry of the system under study here permits some amount of simplification since $\tilde{u}_i = \tilde{u}$, $\hat{\rho}_i = \hat{\rho}$ and $\tilde{p}_i(n) = \tilde{p}(n)$ for all i. This fact may be readily established in a more rigorous manner. Consequently,

$$\hat{p}(n_1,n_2,\ldots,n_M) = \pi(1-\tilde{u})^{M-k}[\tilde{u}(1-\hat{\rho})]^k\hat{\rho}^{N-k}$$

$$= [\pi(1-\tilde{u})^M\hat{\rho}^N][\frac{\tilde{u}(1-\hat{\rho})}{(1-\tilde{u})\hat{\rho}}]^k$$

$$= \pi'\theta^k$$

where k is the number of busy servers,

$$\theta = \frac{\tilde{u}(1-\rho)}{(1-\tilde{u})\rho}$$

and π' is a new normalization constant. However, θ cannot be evaluated since $\tilde{u}$ is unknown and the literature does not provide any satisfactory procedure for calculating it. For this reason, the method of calculating $\hat{\rho}$ is irrelevant. In Section 5.3, a method for directly evaluating θ is developed specifically for the multiprocessor memory bandwidth problem. (If θ is arbitrarily chosen to be 1 then every state is equiprobable and the result obtained in [22] follows).

Overlooking the fact that θ is unknown, one may proceed to obtain results which will be of use in Section 5.3. Clearly, there are $\binom{M}{i}$ ways of selecting i busy modules. For each one of these combinations there exist $\binom{N-1}{i-1}$ ways of assigning the N processors to the i busy modules if the processors are considered indistinguishable [32]. (Since the state of a processor's reference process is given by the module last referenced, all processors queued on a module are, indeed, indistinguishable). Consequently, the number of states corresponding to i busy modules is given by $\binom{M}{i}\binom{N-1}{i-1}$. The probability assigned to each such state is $\pi'\theta^i$ and so, by the definition of π',

$$\pi' \sum_{i=1}^{\infty} \binom{M}{i}\binom{N-1}{i-1}\theta^i = 1 \quad \text{and}$$

$$\pi' = 1/\sum_{i=1}^{\infty} \theta^i \binom{M}{i}\binom{N-1}{i-1}$$

The bandwidth is given by the average number of busy memory modules and so

$$B(M,N) = \pi' \sum_{i=1}^{\infty} i\,\theta^i \binom{M}{i}\binom{N-1}{i-1} = \frac{\sum_{i=1}^{\infty} i\,\theta^i \binom{M}{i}\binom{N-1}{i-1}}{\sum_{i=1}^{\infty} \theta^i \binom{M}{i}\binom{N-1}{i-1}}$$

$$= \frac{\sum_{i=1}^{\infty} \theta^i \binom{M-1}{i-1}\binom{N-1}{i-1}}{\sum_{i=1}^{\infty} \frac{\theta^i}{i} \binom{M-1}{i-1}\binom{N-1}{i-1}} = \frac{\sum_{i=0}^{\infty} \theta^i \binom{M-1}{i}\binom{N-1}{i}}{\sum_{i=0}^{\infty} \frac{\theta^i}{i+1} \binom{M-1}{i}\binom{N-1}{i}} \qquad (1)$$

The probability of a given module being idle may be calculated from the expression for the bandwidth or by noting that there are $\binom{M-1}{i}$ ways of selecting i busy modules from the remaining (M-1) modules and that for each of these there are $\binom{N-1}{i-1}$ ways of assigning the N processors to the i busy modules. Therefore,

$$P[\text{a given module is idle}] = \pi' \sum_{i=1}^{\infty} \theta^i \binom{M-1}{i}\binom{N-1}{i-1}$$

$$= \frac{\sum_{i=0}^{\infty} \theta^i \binom{M-1}{i+1}\binom{N-1}{i}}{\sum_{i=0}^{\infty} \theta^i \binom{M}{i+1}\binom{N-1}{i}} \qquad (2)$$

Similarly,

$$P[\text{two given modules are idle}] = \frac{\sum_{i=0}^{\infty} \theta^i \binom{M-2}{i+1}\binom{N-1}{i}}{\sum_{i=0}^{\infty} \theta^i \binom{M}{i+1}\binom{N-1}{i}} \qquad (3)$$

Using arguments of the same nature one finds that

P[queue length at a given module = k and another given module is idle]

$$= \begin{cases} \pi' \sum_{i=1}^{\infty} \theta^{i} \binom{M-2}{i}\binom{N-1}{i-1} & k = 0 \\ \pi' \sum_{i=1}^{\infty} \theta^{i+1} \binom{M-2}{i}\binom{N-k-1}{i-1} & N > k > 0 \\ \pi' \theta & k = N \end{cases}$$

Hence, the average queue length at a given module given that some other module is idle

$$= \frac{\pi' \left(\sum_{k=1}^{N-1} k \sum_{i=1}^{\infty} \theta^{i+1} \binom{M-2}{i}\binom{N-k-1}{i-1} + N\theta \right)}{\pi' \left(\sum_{k=1}^{N-1} \sum \theta^{i+1} \binom{M-2}{i}\binom{N-k-1}{i-1} + \theta + \sum_{i=1}^{\infty} \theta^{i} \binom{M-2}{i}\binom{N-1}{i-1} \right)}$$

$$= \frac{\sum_{i=1}^{\infty} \theta^{i+1} \binom{M-2}{i} \sum_{k=1}^{N-1} k \binom{N-k-1}{i-1} + N\theta}{\sum_{i=1}^{\infty} \theta^{i+1} \binom{M-2}{i} \sum_{k=1}^{N-1} \binom{N-k-1}{i-1} + \theta + \sum_{i=0}^{\infty} \theta^{i+1} \binom{M-2}{i+1}\binom{N-1}{i}}$$

$$= \frac{\sum_{i=0}^{\infty} \theta^{i+1} \binom{M-2}{i}\binom{N}{i+1}}{\sum_{i=0}^{\infty} \theta^{i+1} \binom{M-2}{i}\binom{N-1}{i} + \sum_{i=0}^{\infty} \theta^{i+1} \binom{M-2}{i+1}\binom{N-1}{i}}$$

$$= \frac{\sum_{i=0}^{\infty} \theta^{i+1} \binom{M-2}{i}\binom{N}{i+1}}{\sum_{i=0}^{\infty} \theta^{i+1} \binom{M-1}{i+1}\binom{N-1}{i}} = \frac{N \sum_{i=0}^{\infty} \theta^{i+1} \binom{M-2}{i}\binom{N-1}{i}/(i+1)}{(M-1) \sum_{i=0}^{\infty} \theta^{i+1} \binom{M-2}{i}\binom{N-1}{i}/(i+1)}$$

$$= N/(M-1) \qquad (4)$$

which is a result that might have been expected. These last three expressions are used to evaluate θ in Section 5.3 whereupon the bandwidth is readily obtained.

3.5. An approximate asymptotic analysis

The method used here is basically the one used by Baskett and Smith [24] under the name of the "binomial approximation" and is asymptotically exact as $M, N \to \infty$. However, it is extended to handle the SFMM for the referencing behavior and more attention is paid to details which, though unimportant for the RIRM, lead to additional inaccuracy with the model assumed here.

Assume that the modules are labelled 0 through M-1. The analysis concentrates upon the queue on module 0 which is, for the most part, treated like an isolated queue. The remaining modules are relevant insofar as they affect the arrival process to module 0. Two assumptions are made in the analysis:

Assumption 1. Given that n of the modules 1 through M-1 are busy (when the system is stationary), any combination of n modules is equally likely to be the set of busy modules,

Assumption 2. The distribution of the number of busy modules in the stationary system is independent of the queue length at module 0. (This assumption, which is obviously untrue for small values of M, N, becomes increasingly acceptable as M, N become very large).

Based on these assumptions, the analysis proceeds through the use of probability generating functions (p.g.f.) [42]. Let h_i be the stationary probability that queue 0 is of length i. (The queue always includes the request being served). Then the p.g.f. H(z) is defined to be

$$H(z) = \sum_{i=0}^{N} h_i z^i .$$

Two additional sets of p.g.f.'s are $A_j(z)$ and $B_j(z)$ which correspond to the number of arrivals to queue 0 at the start of a cycle given that j modules in the entire system are busy under two conditions respectively - that module 0 is one of the j busy modules or that module 0 is idle. If q_j is the stationary probability of j busy modules then define

$$A(z) = \sum_{i=1}^{\min(M,N)} q_j A_j(z)$$

and

$$B(z) = \sum_{i=1}^{\min(M,N)} q_j B_j(z).$$

By assumption 2, q_j is independent of the length of queue 0.

If the queue length at module 0 is non-zero, then at the end of a cycle it will be decremented by one with probability $(1-p_0)$ and then incremented by a random variable which is the number of arrivals. If the queue length were 0, it would merely be incremented by the random number of arrivals. Bearing in mind that the addition (subtraction) of an independent random variable corresponds, in the transform domain, to multiplication (division) by the p.g.f. of the random variable we have that the p.g.f. for the queue length during the next cycle is

$$H^+(z) = p_0 [H(z) - H(0)]\, A(z) + (1-p_0)\, \frac{[H(z)-H(0)]}{z}\, A(z)$$

$$+ H(0)B(z)$$

$$= [H(z) - H(0)]\,(p_0 + \frac{1-p_0}{z})\, A(z) + H(0)B(z)$$

$$= H(z) \text{ by stationarity.}$$

Note that $H(0) = h_0$ = the probability that queue 0 is empty. The above equation may be rewritten by collecting all terms involving $H(z)$ on the left hand side and then dividing on both sides by the coefficient of $H(z)$ to obtain

$$H(z) = H(0)\,\frac{zB(z) - (1-p_0+p_0z)A(z)}{z-(1-p_0+p_0z)A(z)}$$

The strategy is to solve for $H(0)$, the idle fraction, by evaluating the equation (or some derivative of it) at the point $z=0$ or 1 without obtaining a trivial identity. It turns out that setting $z=0$ or 1 in the above equation yields obvious identities. However, if we differentiate both sides with respect to z and set z equal to 1, we obtain $H'(1)$ on the left hand side, which is the mean length of queue 0 and this, by symmetry, must be equal to N/M. The evaluation of the right hand sides requires two successive applications of L'Hospitals' rule and after some simple,if tedious,computation yields

$$H'(1) = \frac{N}{M} = H(0)\,\frac{[1-p_0-A'(1)][2B'(1)+B''(1)]+B'(1)[2p_0A'(1)+A''(1)]}{2(1-p_0-A'(1))^2} \tag{5}$$

The idle fraction is obtained if $A'(1)$, $A''(1)$, $B'(1)$ and $B''(1)$ are known.

Let $\mathcal{C}_n$ be an ordered set of all combinations obtained by selecting n modules out of the modules 1 through M-1. Let c_k be the k-th such combination in $\mathcal{C}_n$ and let $\{k_1,\ldots,k_n\}$ be the indices of the n selected modules. Then if i is the number of busy modules in the system we have

$$A_i(z) = \frac{\displaystyle\sum_{c_k\in\mathcal{C}_{i-1}} \prod_{j=1}^{i=1}(1-p_{-k_j}+p_{-k_j}z)}{\binom{M-1}{i-1}} \qquad \text{and}$$

$$B_i(z) = \frac{\sum_{c_k \in \mathcal{C}_i} \prod_{j=1}^{i} (1-p_{-k_j} + p_{-k_j} z)}{\binom{M-1}{i}}$$

where p_{-k_i} is the probability of a processor referencing module 0 after referencing module k_i. (The negative index denotes the positive index obtained by performing the negation modulo M). These expressions depend on the assumption that any combination of modules is equally likely to be the set of busy modules. In writing down $A_i(z)$, the fact that module 0 is busy has been accounted for by selecting combinations of (i-1) modules from the remaining (M-1) modules. This assumes that the total number of busy modules is independent of the queue length at module 0. Now,

$$A_i'(1) = \sum_{c_k \in \mathcal{C}_{i-1}} \sum_{j=1}^{i=1} p_{-k_j} / \binom{M-1}{i-1} \qquad \text{and}$$

$$A_i''(1) = \sum_{c_k \in \mathcal{C}_{i-1}} \sum_{j=1}^{i-1} \sum_{\substack{r=1 \\ r \neq j}}^{i-1} p_{-k_j} p_{-k_r} / \binom{M-1}{i-1}$$

To evaluate $A_i'(1)$ further note that of the $\binom{M-1}{i-1}$ possible combinations, module t appears in exactly $\binom{M-2}{i-2}$ combinations and consequently p_{-t} will appear in the summation $\binom{M-2}{i-2}$ times. Therefore,

$$A'_i(1) = \sum_{j=1}^{M-1} p_j \binom{M-2}{i-2} / \binom{M-1}{i-1} = \frac{(i-1)}{(M-1)} \sum_{j=1}^{M-1} p_j = \frac{(i-1)}{(M-1)} (1-p_0)$$

and by similar arguments

$$A_i''(1) = \frac{(i-1)(i-2)}{(M-1)(M-2)} \sum_{j=1}^{M-1} \sum_{\substack{r=1 \\ r\neq 1}}^{M-1} p_j p_r$$

$$= \frac{(i-1)(i-2)}{(M-1)(M-2)} [1 - 2p_0 + 2p_0^2 - \sum_{j=0}^{M-1} p_j^2]$$

$$= \frac{(i-1)(i-2)}{(M-1)(M-2)} \cdot \; 2(1-p_0)(\frac{1}{\gamma} - p_0)$$

where $\gamma = 2(1-p_0)/(1 - \sum_{j=0}^{M-1} p_j^2)$

Therefore,

$$A'(1) = \sum_{i=1}^{\min(M,N)} q_i \, A_i'(1) = \frac{(1-p_0)}{(M-1)} \sum_{i=1}^{\min(M,N)} (i-1)q_i$$

$$= \frac{(1-p_0)}{(M-1)} (\mu - 1) \qquad \text{and}$$

$$A''(1) = \frac{2(1-p_0)(1/\gamma - p_0)}{(M-1)(M-2)} \sum_{i=1}^{\min(M,N)} (i-1)(i-2)q_i$$

$$\frac{2(1-p_0)(1/\gamma - p_0)}{(M-1)(M-2)} [\nu + (\mu - 1)(\mu - 2)]$$

where μ is the mean number of busy modules in the system and ν is the variance of this number. Similarly,

$$B'(1) = \frac{(1-p_0)}{(M-1)} \mu \qquad \text{and}$$

$$B''(1) = \frac{2(1-p_0)(1/\gamma - p_0)}{(M-1)(M-2)} [\nu + \mu(\mu - 1)].$$

Now, as in [24], one may argue that as M and N both become extremely large, the various queues become increasingly independent of one another. Consequently, the central limit theorem applies to the distribution of the number of busy modules. In particular, the quantity $\nu/\mu^2 \rightarrow 0$ and, hence,

$$A''(1) \rightarrow \frac{2(1 - p_0)(1/\gamma - p_0)}{(M-1)(M-2)}(\mu - 1)(\mu - 2) \qquad \text{and}$$

$$B''(1) \rightarrow \frac{2(1 - p_0)(1/\gamma - p_0)}{(M-1)(M-2)}\mu(\mu - 1).$$

Equation 5 may now be used to solve for the value of μ by substituting the (asymptotic) values of $A'(1)$, $A''(1)$, $B'(1)$ and $B''(1)$ and noting that

$$H(0) = 1 - \mu/M$$

to obtain

$$B(M,N,\gamma) = \mu = \frac{\gamma(M+N) - 1 \pm \sqrt{[\gamma(M+N) - 1]^2 - 4\gamma(\gamma - 1)MN}}{2(\gamma - 1)} \tag{6}$$

the appropriate sign being chosen to ensure that μ is not greater than either M or N.

A few special cases are of interest. For the RIRM, γ is equal to to 2 and the expression reduces to that obtained by Baskett and Smith via the "binomial approximation". As $p_0 \rightarrow 1$, $\gamma \rightarrow 1$ and the queueing system tends to an exponential queueing network. This is clear if one views a sequence of successive visits of unit service time to a module as constituting one visit with a geometrically distributed service time. As $p_0 \rightarrow 1$, the mean service time becomes exceedingly greater than one cycle and if the distribution function is scaled so that the mean is one, then the stepwise increasing function begins to appear continuous and exponential. For a value of $\gamma = 1$,

$$B(M,N,1) = \frac{MN}{M+N-1}$$

which is the formula derived in Section 3.3 for the exponential model. As $p_0 \to 0$ and $p_i \to 1$ for some $i \neq 0$, $\gamma \to \infty$ and

$$B(M,N,\infty) = \min(M,N).$$

This is the expected value for the bandwidth since all the processors get synchronized. The range of variation of γ is between 1 and ∞ and μ is an increasing function of γ. At the two extreme values of γ, the value of μ is exact. However, at intermediate points such as $\gamma = 2$, it is fairly accurate but not exact. Also, since the assumptions made are correct for $M,N \to \infty$, the expression is asymptotically correct.

4. An exact analysis

The analysis in this section makes use of the probability generating function approach and bears some resemblance to the analysis of Section 3.5. However, it makes no approximations. The state of the system during any cycle is given by the vector $\langle n_0, \ldots, n_{M-1} \rangle$ which lists the number of requests queued at each module. The set of all feasible states constitutes a Markov chain since the probability distribution of the next state depends only on the current state. Furthermore, this Markov chain is aperiodic since it is possible to make a one-step transition from any state back to the same state, and it is irreducible since it is possible to reach any state from any other state in a finite number of transitions. Hence, the Markov chain is ergodic and possesses a unique stationary probability distribution.

Let $\mathscr{S}$ be the class of all feasible states (for which $n_0+\ldots+n_{M-1} = N$). Let $P(S_j)$ be the stationary probability of being in state S_j and let the corresponding vector be $\langle n_{0j},\ldots,n_{M-1,j}\rangle$. Then, define the probability generating function

$$H(Z) = H(z_0,\ldots,z_{M-1}) = \sum_{S_j\in\mathscr{S}} p(S_j) \prod_{i=0}^{M-1} z_i^{n_{ij}}.$$

Let $Q_i(Z)$ be the p.g.f. for the routing probabilities of a processor whose last request was to module i, i.e.,

$$Q_i(Z) = Q_i(z_0,\ldots,z_{M-1}) = \sum_{j=0}^{M-1} p_{j-i} z^j$$

where the subtraction in the subscript is performed modulo M.

At the end of a cycle, each module which was not idle, releases a serviced processor which is then reassigned to some module based on the routing probabilities. Define R_i to be the operator which operates on $H(Z)$ to yield the p.g.f. of the system after module i has released a processor (if not idle). Just as z_i is the dummy variable associated with processors enqueued at module i, let y_i be the dummy variable associated with a processor that has just been released by module i. Lastly, if U_i is defined to be the operator which sets $z_i = 0$ in $H(Z)$ then, just as in Section 3.5,

$$\begin{aligned} R_iH(Z) &= U_iH(Z) + [H(Z) - U_iH(Z)]\,\frac{y_i}{z_i} \\ &= \frac{y_i}{z_i}\,H(Z) + \left(1 - \frac{y_i}{z_i}\right)U_iH(Z). \end{aligned}$$

From the above expression one may define

$$R_i = [\frac{y_i}{z_i} + (1 - \frac{y_i}{z_i}) \{U_i\}]$$

Note that U_i and U_j, $(i \neq j)$, are commutative operations since they operate on distinct dummy variables. Therefore, at the end of a cycle, after each module has released a processor (if any), the p.g.f. for the system is

$$\{\prod_{i=0}^{M-1} R_i\} H(Z) = \{\prod_{i=0}^{M-1} [\frac{y_i}{z_i} + (1 - \frac{y_i}{z_i}) U_i]\} H(Z).$$

In the transform domain, the processors are reassigned by substituting $Q_i(Z)$ for y_i in the above expression to obtain the p.g.f. for the system during the next cycle. By stationarity this must be equal to H(Z) and, therefore,

$$H(Z) = \{\prod_{i=0}^{M-1} [Q_i(Z)/z_i + (1 - Q_i(Z)/z_i)U_i]\} H(Z). \qquad (7)$$

This equation is exactly equivalent to equating the stationary probability vector for the underlying Markov chain to the product of the same vector and the transition matrix. However, it does not require the enumeration of all the states and is consequently more convenient. It also permits the analysis to proceed in a much smoother fashion.

Let D_i be the operator that differentiates with respect to z_i. Then, bearing in mind that

$$D_i(U_j H(Z)) = \begin{cases} U_j(D_i H(Z)) & i \neq j \\ 0 & i = j \end{cases}$$

one observes that

$$D_i \{[Q_j(Z)/z_j + (1 - Q_j(Z)/z_j)U_j] H(Z)\}$$

$$= \{[D_i(Q_j(Z)/z_j) + (Q_j(Z)/z_j)D_i$$

$$+ [D_i(1 - Q_j(Z)/z_j)]U_j + (1 - Q_j(Z)/z_j)D_iU_j \}H(Z)$$

By the symmetry of the problem, the marginal queue length distributions for all modules are identical and it is necessary to study only one, the one for module 0. The p.g.f. of this marginal distribution is obtained by setting $z_1 = \ldots = z_{M-1} = 1$. For notational convenience z_0 is set equal to z and the following definitions are made:

given that $Z^* = \{z,1,\ldots,1\}$ define

$h(z) = H(Z^*)$

$V_i(z) = Q_i(Z^*)$

$u_0h(z) = h(0)$

$U_ih(z) = U_iH(Z_{(i)}) \qquad i \neq 0$

where $Z_{(i)}$ corresponds to $z_0 = z$, $z_i = 0$ and $z_j = 1 (j \neq 0,i)$.

Equation 7 now assumes the form

$$h(z) = \{[\frac{V_0(z)}{z} + (1 - \frac{V_0(z)}{z}) U_0] \prod_{i=1}^{M-1} [V_i(z) + (1 - V_i(z))U_i] \quad h(z).$$

The right hand side includes a term in h(z) with the coefficient $(1/z) \prod_{i=0}^{M-1} V_i(z)$. Collecting terms in h(z) on the left hand side and the dividing on both sides by the coefficient of h(z) one obtains

$$h(z) = \frac{\{[V_0(z)+(z-V_0(z)U_0] \prod_{i=1}^{M-1} [V_i(z)+(1-V_i(z))U_i] \prod_{i=0}^{M-1} V_i(z)\}h(z)}{z - \prod_{i=0}^{M-1} V_i(z)}$$

The most convenient and non-trivial identity is obtained by putting $z = 1$. This entails applying L'Hospital's rule twice on the right hand side and the rules regarding the commutativity of D_0 and U_0 (or V_0) must be borne in mind. The result of setting $z = 1$ is to get $h(1) = 1$

$$= \frac{\left\{ \begin{array}{l} 2 \sum_{i=1}^{M-1} V_i'(1) U_i h'(1) - \sum_{i=0}^{M-1} \sum_{\substack{j=0 \\ j \neq i}}^{M-1} V_i'(1) V_j'(1) [1 - U_i - U_j + U_i U_j] h(1) \\ -2 \sum_{i=1}^{M-1} V_i'(1) [U_0 - U_0 U_i] h(1) + 1 - \sum_{i=0}^{M-1} [V_i'(1)]^2 \end{array} \right\}}{1 - \sum_{i=0}^{M-1} [V_i'(1)]^2} \qquad (8)$$

where $V_i'(1) = p_{-i}$

$U_j h(1) = P[\text{module } j \text{ is idle}]$

$U_j h(1) = U_j h(1)$ for all i,j by symmetry

$U_i U_j h(1) = P[\text{module } i \text{ and module } j \text{ are idle}]$

$= P[\text{module } i \text{ is idle} \mid \text{module } j \text{ is idle}] . P[\text{module } j \text{ is idle}]$

and $U_i h'(1) = [\text{Mean queue length at module } 0 \mid \text{module } i \text{ is idle}] .$
$P[\text{module } i \text{ is idle}]$

Our interest is in solving for $U_0 h(1)$ but due to the presence of terms of the form $U_i U_j h(1)$ and $U_i h'(1)$ we cannot do so immediately.

5. "Almost-exact" solutions

The analysis of the proceeding section can continue in certain special cases by making assumptions which are intuitively plausible:

5.1. The RIRM case

Define $F(M,N)$ to be the marginal probability of a module being idle in a system with M modules and N processors. Therefore, $F(M,N) = U_i h(1)$ for all i. The approximation introduced in [27] is to assume that the marginal queue length distribution at a module in a system with M modules, given that some other module is idle, is identical to the marginal queue length distribution in a system with only M-1 modules. The use of a slightly stronger approximation has been suggested in [24]. As a consequence of this assumption,

$$U_i U_j h(1) = F(M-1,N)F(M,N) \quad \text{and}$$

$$U_i h'(1) = \frac{N}{(M-1)} \cdot F(M,N)$$

and Equation 8 yields

$$F(M,N) = \frac{1}{\frac{2N}{M-1} + F(M-1,N)} .$$

This recurrence may be solved by noting that $F(1,N) = 1$ for all $N \geq 1$. In [27] it is shown that the recurrence is satisfied by the following solution:

$$F(M,N) = 1 - \frac{B(M,N)}{M}$$

where the memory bandwidth B(M,N) is given by

$$B(M,N) = \frac{\sum_{i=0}^{\infty} z^i \binom{M-1}{i} \binom{N-1}{i}}{\sum_{i=0}^{\infty} \frac{z^i}{(i+1)} \binom{M-1}{i} \binom{N-1}{i}} .$$

Note that the above summations are finite since the terms become 0 for $i \geq M$ or N. This expression has been compared to the exact results listed in [22]. The maximum error was found to occur for values of $M = N$. Table 5 compares the error of Strecker's approximation, the Binomial approximation and the above expression for values of $M = N$. The error of this "almost-exact" solution is always less than 0.25%.

5.2. The Localized Reference Model

The assumption made for the RIRM analysis seems equally justified when analyzing the localized model of Sethi and Deo. Let $p_0 = \alpha$. Then from Equation 8 one obtains [27].

$$F(M,N,\alpha) = \frac{1}{\frac{\gamma N}{M-1} + 2 - \gamma + (\gamma-1)F(M-1,N)}$$

and

$$B(M,N,\alpha) = \frac{\sum_{i=0}^{\infty} \gamma^i \binom{M-1}{i}\binom{N-1}{i}}{\sum_{i=0}^{\infty} \frac{\gamma^i}{(i+1)} \binom{M-1}{i}\binom{N-1}{i}}$$

where $$\gamma = \frac{2(M-1)}{(1+\alpha)M-2}$$

This result cannot be directly tested for accuracy since no exact analysis has been performed. However, Sethi and Deo [26] have performed simulation runs each of length 5000 cycles for values of N and M ranging from 2 through 8 and 2 through 10 respectively. For each value of (M,N) they have tabulated the bandwidth averaged over values of $\alpha = 0.75$, 0.8, 0.85, 0.9 and 0.95. Table 6 displays these values for $M = N$ and compares them with corresponding values obtained via the above expression. The percentage difference, too, is tabulated. This difference is the combined effect of the error in the analysis as well as the statistical error in the simulations. The difference is less than 1% over the range of M and N examined.

5.3. The Symmetric First-order Markov Model

The assumption made for the RIRM and the LRM in Sections 5.1 and 5.2 respectively cannot be accepted quite as readily in the case of the SFMM. The system with one less module will have a different transition matrix which cannot be easily related to that of the original system. However, if one is willing to accept the diffusion approximation then the exact analysis of Section 4 can be continued to provide an approximate result.

Using the terminology of Section 4 and the results of Section 3.4 (Equations 2-4) the following equations are obtained:

$$U_j h(1) = \frac{\sum_{i=0}^{\infty} \theta^i \binom{M-1}{i+1}\binom{N-1}{i}}{\sum_{i=0}^{\infty} \theta^i \binom{M}{i+1}\binom{N-1}{i}} \overset{\Delta}{=} \frac{A}{D} \text{ for all } j$$

$$U_k U_j h(1) = \frac{\sum_{i=0}^{\infty} \theta^i \binom{M-2}{i+1}\binom{N-1}{i}}{\sum_{i=0}^{\infty} \theta^i \binom{M}{i+1}\binom{N-1}{i}} \overset{\Delta}{=} \frac{B}{D} \text{ for all } k,j$$

$$U_i h'(1) = \frac{N}{M-1} \text{ for all } i$$

$$\sum_{i=0}^{M-1} V_i'(1) = 1$$

$$\sum_{i=1}^{M-1} V_i'(1) = 1-p_0$$

$$\sum_{i=0}^{M-1} [V_i'(1)]^2 = \sum_{i=0}^{M-1} p_i^2$$

$$\sum_{i=0}^{M-1} \sum_{\substack{j=0 \\ j\neq i}}^{M-1} V_i'(1)V_j'(1) = 1 - \sum_{i=0}^{M-1} p_i^2$$

Then, from Equation 8,

$$1-\sum_{i=0}^{\infty} p_i^{\,2} = 2(1-p_0)\,\frac{N}{M-1}\,\frac{A}{D}$$

$$+\,2(1-\sum_{i=0}^{\infty} p_i^{\,2} - 1 + p_0)\frac{A}{D}$$

$$+\,[2(1-p_0) - (1-\sum_{i=0}^{\infty} p_i^{\,2})]\frac{B}{D}$$

Substituting γ for $2(1-p_o)/(1-\sum_{i=0}^{\infty}p_i^{\,2})$ one gets

$$(\frac{\gamma N}{M-1} + 2 - \gamma)A + (\gamma - 1)B = D$$

Now,

$$\frac{\gamma N}{M-1}A - \gamma A + \gamma B = \gamma\sum_{i=0}^{\infty}\theta^i[\binom{M-2}{i}\binom{N}{i+1} - \binom{M-1}{i+1}\binom{N-1}{i} + \binom{M-2}{i+1}\binom{N-1}{i}]$$

$$= \gamma\sum_{i=0}^{\infty}\theta^i[\binom{M-2}{i}\binom{N-1}{i+1} + \binom{M-2}{i}\binom{N-1}{i} - \binom{M-2}{i}\binom{N-1}{i}]$$

$$= \gamma\sum_{i=0}^{\infty}\theta^i\binom{M-2}{i}\binom{N-1}{i+1}$$

and

$$D - 2A + B = \sum_{i=0}^{\infty}\theta^i[\binom{M}{i+1}\binom{N-1}{i} - 2\binom{M-1}{i+1}\binom{N-1}{i} + \binom{M-2}{i+1}\binom{N-1}{i}]$$

$$= \sum_{i=0}^{\infty}\theta^i[\binom{M-1}{i}\binom{N-1}{i} - \binom{M-2}{i}\binom{N-1}{i}]$$

$$= \sum_{i=0}^{\infty}\theta^i\binom{M-2}{i-1}\binom{N-1}{i}$$

$$= \sum_{i=1}^{\infty}\theta^i\binom{M-2}{i-1}\binom{N-1}{i} \text{ since } \binom{M-2}{-1} = 0$$

$$= \sum_{i=0}^{\infty}\theta^{i+1}\binom{M-2}{i}\binom{N-1}{i+1}$$

Therefore $\theta = \gamma$ and the bandwidth

$$B(M,N,\gamma) = \frac{\sum_{i=0}^{\infty} \gamma^i \binom{M-1}{i}\binom{N-1}{i}}{\sum_{i=0}^{\infty} \frac{\gamma^i}{i+1} \binom{M-1}{i}\binom{N-1}{i}}$$

from Equation 1. This expression is termed the <u>Diffusion/Generating Function (DGF)</u> solution.

It is easily ascertained that $B(M,N,\gamma)$ satisfies the recurrence

$$B(M,N,\gamma) = M. \frac{\gamma N - (\gamma - 1)B(M-1,N,\gamma)}{\gamma N + M - 1 - (\gamma - 1)B(M-1,N,\gamma)}.$$

This recurrence may be used to derive the asymptotic result of Section 3.5. For large values of M it might be expected that the addition of another module would not have an appreciable effect upon the bandwidth, i.e., $B(M-1,N,\gamma) \simeq B(M,N,\gamma)$. The use of this approximate equality converts the recurrence relation into a quadratic equation in $B(M,N,\gamma)$, the solution of which yields Equation 6.

The expression for $B(M,N,\gamma)$ is symmetric in M and N when γ is fixed. However, since γ may be expected to vary with M, the bandwidth observed in a system, in which all processors have identical reference behavior, will not be symmetric in M and N.

The expression for $B(M,N,\gamma)$ is exact if $\gamma = 1$ or if $\gamma = \infty$. For $\gamma = 2$, the result of Section 5.1 for the RIRM case is obtained. This was found to be accurate within a quarter of one percent. To assess the error of this expression for an arbitrary value of γ, four simulations were performed corresponding to those described in Section 2 (N=4 and M=2,4,8,16). The measured bandwidth was compared with the bandwidth computed analytically using the value of γ obtained from the estimated transition probabilities (Table 4). The discrepancies may be attributed to four factors:

1) the SFMM is not an adequate model of program behavior due to the presence of higher order dependencies,
2) the reference patterns of the four processors are not statistically identical as assumed by the model,
3) the statistical error in the simulation,
4) the approximations introduced in the analysis.

Table 7 displays the bandwidths obtained from the simulation, the DGF solution and the asymptotic solution. The percentage errors are also shown in parentheses. The agreement is seen to be quite satisfactory.

In an attempt to separate the error caused by the analysis from that due to assumptions in the model, two additional simulations were performed for N=4 and M=2,4. Of the four factors listed above, the first two were eliminated by generating reference strings which conformed exactly with the model, i.e. all four reference strings were statistically identical and were generated using the SFMM parameters of Table 4. Both runs were 2500 cycles long. Table 8 records the measured bandwidth, the analytical prediction and the percentage error. In conjunction with Table 7 one may conclude that a major portion of the error is due to the fact that the SFMM model is not quite adequate in describing real reference strings. However, absolutely speaking, the error caused by modelling approximations is acceptably small. The error due to the analysis itself (Table 8) is smaller still. (Note that the recorded error also includes the statistical simulation error).

6. A Two-parameter model

The Symmetric First-order Markov Model would appear to be an adequate model of program behavior, but it has one drawback in that a separate model is needed for each degree of interleaving. It is clearly desirable that the specification of the model of program behavior be independent of the physical structure of the memory since, firstly, the program exists regardless of the memory and, secondly, greater parsimony and generalityare achieved by having just one model which is applicable to any memory structure of interest.

To obtain a memory interleaved 8 ways from one interleaved 16 ways, module i is combined with module $i+8$ (modulo 16). This assumes low-order interleaving of the modules. Correspondingly, states i and $i+8$ must be combined in the Markov chain which describes program behavior. It is neccessary to establish whether the resulting lumped chain has the Markov property. If so, this permits the derivation of the model of program behavior for $M=8$ from the for $M=16$.

In the general case one has $M(>2)$ modules (M, being a power of 2 and the associated SFMM parameters $p_0(M)$ through $p_{M-1}(M)$. A partition $A=\{A_0,\ldots.,A_m\}$ is defined upon the state space of the SFMM whereby state i is mapped into A_j where $j=i \bmod m$ and m is assumed to be a power of 2. The Markov chain with state space $\{0,\ldots,M-1\}$ is said to be lumpable with respect to the partition A if each equivalence class corresponds to a state in a Markov chain with a reduced state space. It can be shown [43] that a Markov chain is lumpable with respect to a partition A if and only if for any $x,y \in A_i$,

$$q^*_{ij} = p[\text{next state} \in A_j / \text{current state is } x]$$

$$= p[\text{next state} \in A_j / \text{current state is } y] \text{ for all}$$

A_i, $A_j \in A$. Furthermore, if this condition is satisfied q^*_{ij} is the conditional probability of a transition to A_j given that the current state is in A_i. It is obvious by inspection of the SFMM transition matrix that this condition is satisfied and that q^*_{ij} as defined above will be given by

$$q^*_{i,i+j} = \sum_{k=0}^{r-1} p^{(M)}_{km+j}$$

where $r = M/m$ and $M > 2$ is a power of 2. By defining $p_j(m)$ to be equal to $q^*_{i,i+j}$ one obtains the SFMM parameters for a degree of interleaving of m. Hence, the SFMM parameters corresponding to the maximum degree of interleaving of interest can be used to derive the SFMM parameters for any lesser degree of interleaving.

The reverse procedure for obtaining the parameters for 2M modules from those for M modules does not exist due to an excess of unknowns over equations. However, an examination of Table 4 indicates that it is reasonable to assume that

$$p_i(M) = p_j(M) \text{ for } i,j \neq 0 \text{ or } 1, \text{ i.e.,}$$

$$p_i(M) = \frac{1-p_0(M)-p_1(M)}{M-2}, \; i \neq 0 \text{ or } 1,$$

and that

$$p_i(2M) = \frac{1-p_0(2M)-p_1(2M)}{2M-2} \; .$$

Then, if $p_0(M)$ and $p_1(M)$ are obtained from the parameters for 2M modules one gets

$$p_0(M) = p_0(2M) + \frac{1-p_0(2M)-p_1(2M)}{2M - 2}$$

$$p_1(M) = p_1(2M) + \frac{1-p_0(2M)-p_1(2M)}{2M - 2} .$$

$p_0(2M)$ and $p_1(2M)$ may be solved for to obtain

$$p_0(2M) = p_0(M) - \frac{1-p_0(M)-p_1(M)}{2(M-2)}$$

$$p_1(2M) = p_1(M) - \frac{1-p_0(M)-p_1(M)}{2(M-2)} .$$

Noting that $[1-p_0(2M)-p_1(2M)] = [1-p_0(M)-p_1(M)]\ (M-1)/(M-2)$ one may repeat this process to obtain

$$p_0(4M) = p_0(M) - \frac{3}{4} \cdot \frac{1-p_0(M)-p_1(M)}{(M-2)}$$

$$p_1(4M) = p_0(M) - \frac{3}{4} \cdot \frac{1-p_0(M)-p_1(M)}{(M-2)} .$$

If one repeats this process indefinitely, one gets

$$\alpha = p_0(\infty) = p_0(M) - \frac{1-p_0(M)-p_1(M)}{(M-2)}$$

$$\beta = p_1(\infty) = p_1(M) - \frac{1-p_0(M)-p_1(M)}{(M-2)} .$$

α and β may be interpreted as the probabilities of referencing the same word and the sequentially following word respectively. For any M,

$$p_i(M) = \begin{cases} \alpha + \frac{1-\alpha-\beta}{M}, & i=0 \\ \beta + \frac{1-\alpha-\beta}{M}, & i=1 \\ \frac{1-\alpha-\beta}{M}, & i \neq 0,\ 1. \end{cases}$$

Thus, the two-parameter model of program behavior assumes that each reference with probability α references the same module as the last reference and with probability β it references the sequentially following module. If neither of these events occur then the reference is random and equally likely to be to any one of the M modules. Table 9 displays the values of α and β as calculated from the SFMM parameters for $M=4,8$ and 16. Table 10 lists the percentage errors incurred by using the first and third pairs of values for α and β ($M=4$ and $M=16$) to calculate memory bandwidth, the simulation results being used to effect the comparison. The close agreement serves to establish the validity of the two-parameter model. Better results are obtained when α and β are estimated via the SFMM parameters for larger values of M.

7. Conclusion

A study of the memory referencing behavior of four programs indicates that the often used Random Independent Reference Model of program behavior is not valid. The Independent Reference Model and the Localized Reference Model were likewise ruled out as inadequate. The Symmetric First-order Markov Model was proposed as a realistic model of the meory reference string and various approximate but accurate analyses were performed upon it to yield expressions for the memory bandwidth. It was also demonstrated that the model of program behavior could be further simplified to obtain a two-parameter model without sacrificing accuracy appreciably.

Acknowledgements

The trace tapes used in this study are part of a trace library created at RCA. These tapes were acquired by the author through a chain consisting of R. O. Winder of RCA, T. Bonn of Sperry-Univac and G. E. Rossmann of Palyn Associates. Most of the computer time was made available by the Stanford Linear Accelerator Center. The simulations for Table 8 were performed by Walter Osterman.

References

1. IBM System/370 Model 158 Maintenance Diagram Manual, Manual No. SY-22-6912-1, IBM Corp., Armonk, NY.

2. IBM System/370 Model 168 Theory of Operation/Diagram Manual, Processor Storage Control Function, Vol. 4, Manual No. SY-22-6934-1, IBM Corp., Armonk, NY.

3. Amdahl 470 V/6 machine reference, Amdahl Corp., Sunnyvale, Calif., 1975.

4. Conti, C. J., Gibson, D. H., and Pitkowsky, S. H., "Structural aspects of the System/360 Model 85, I. General Organization", IBM Sys. Jour., 7, 1, 1968.

5. PDP-11/70 Processor Handbook, Digital Equipment Corp., 1975.

6. Belady, L. A., "A Study of Replacement Algorithms for a Virtual Storage Computer", IBM Sys. Jour., 5, 2, 1966.

7. Kilburn, T., Edwards, D. B. G., Lanigan, M. J., and Sumner, F. H., "One level storage system", IRE Trans., EC-11, 2, 223-235, April 1962.

8. Meade, R. M., "On memory system design", Proc. FJCC, 1970.

9. Kaplan, K. R., and Winder, R. O., "Cache based computer systems", Computer, 6,3, March 1973.

10. Bell, J., Casasent, D., and Bell, C. G., "An investigation of alternative cache organizations", IEEE Trans. Comput., C-23, 4, April 1974.

11. Strecker, W. D., "Cache memories for PDP-11 Family computers", Proc. Third Annual Symp. Comp. Arch., pp. 155-158, Jan. 1976.

12. Flores, I., "Derivation of a waiting-time factor for a multiple-bank memory", Jour. Assoc. Comput. Mach., 11, 3, pp. 265-282, July 1964.

13. Hellerman, H., Digital Computer System Principles, McGraw-Hill, 1967, pp. 228-229.

14. Burnett, G. J., and Coffman, E. G., "A study of interleaved memory systems", Proc. SJCC, 1970, pp. 467-474.

15. Coffman, E. G., Burnett, G. J., and Snowdon, R. A., "On the performance of interleaved memories with multiple-word bandwidth", IEEE Trans. Comput., C-20, 12, pp. 1570-1573, Dec. 1971.

16. Burnett, G. J., and Coffman, E. G., "Analysis of interleaved memory systems using blockage buffers", Comm. Assoc. Comput. Mach., 18, 2, pp. 91-95, Feb. 1975.

17. Knuth, D. E., and Rao, G. S., "Activity in an interleaved memory", IEEE Trans. Comput., C-24, 9, pp. 943-944, Sep. 1975.

18. Terman, F. W., "A study of interleaved memory systems by trace driven simulation", Proc. Fourth Symp. Simul. Computer Sys., Aug. 1976.

19. Rau, B. R., "Program behavior and the performance of interleaved memories", IEEE Trans. Comput. (to appear).

20. Skinner, C., and Asher, J., "Effect of storage contention on system performance", IBM Sys. Jour., 8, 4, pp. 319-333, 1969.

21. Strecker, W. D., "Analysis of the instruction execution rate in certain computer structures", Ph.D. Thesis, Carnegie-Mellon U., Pittsburgh, PA, 1970.

22. Bhandarkar, D. P., and Fuller, S. H., "Markov chain models for analyzing memory interference in multiprocessor computer systems", Proc. First Annual Symp. on Comp. Arch., pp. 1-6, December 1973.

23. Bhandarkar, D. P., "Analysis of memory interference in multiprocessors", IEEE Trans. Comput., C-24, 9, pp. 897-908, Sept. 1975.

24. Baskett, F., and Smith, A. J., "Interference in multiprocessor computer systems with interleaved memory", Comm. Assoc. Comput. Mach., 19, 6, pp. 327-334, June 1976.

25. Hoogendoorn, C. H., "A general model for memory interference in multiprocessors", IEEE Trans. Comput., C-26, 10, pp. 998-1005, Oct. 1977.

26. Sethi, A. S., and Deo, N., "Interference in multiprocessor systems with varying memory access probabilities", IEEE Computer Society Repository No. R77-220, 1977.

27. Rau, B. R., "Program behavior and the performance of memory systems", Ph. D. Thesis, Stanford U., Stanford, CA, July 1977.

28. Briggs, F. A., and Davidson, E. S., "Organization of LSI semiconductor memories for parallel-pipelined processors", IEEE Trans. Comput., C-26, 2, pp. 162-169, Feb. 1977.

29. Wulff, W., and Bell, C. G., "C.mmp, a multi-mini-processor", Proc. FJCC, 1972, Vol. 41, part II, pp. 765-777, 1972.

30. Winder, R. O., "A data base for computer performance evaluation", Computer, 6, 3, pp. 25-29, March 1973.

31. Rau, B. R., "Sequential prefetch strategies for instructions and data", (submitted for publication). Also in reference 27.

32. Liu, C. L., Introduction to Combinatorial Mathematics, McGraw-Hill, 1968.

33. Gordon, W. J., and Newell, G. S., "Closed queueing systems with exponential servers", Op. Res., Vol. 15, pp. 254-265, 1967.

34. Feller, W., An Introduction to Probability Theory and its Applications, Vol. II, Wiley, New York, 1966.

35. Gaver, D. P., "Diffusion approximations and models for certain congestion problems", Jour. App. Prob., Vol. 5, pp. 607-623, 1968.

36. Baskett, F., Chandy, K. M., Muntz, R. R., and Palacios, F. G., "Open, closed, and mixed networks of queues with different classes of customers", Jour. Assoc. Comput. Mach., 22, 2, pp. 248-260, April 1975.

37. Chandy, K. M., Howard, J. H., and Towsley, D. F., "Product form and local balance in queueing networks", Jour. Assoc. Comput. Mach., 24,2, pp. 250-263, April 1977.

38. Kobayashi, H., "Application of the diffusion approximation to queueing networks, I: Equilibrium queue distributions", Jour. Assoc. Comput. Mach., 21, 2, pp. 316-328, April 1974.

39. Reiser, M., and Kobayashi, H., "Accuracy of the diffusion approximation for some queueing systems", IBM Jour. Res. Develop., pp. 110-124, March 1974.

40. Gelenbe, E., "On approximate computer system models", Jour. Assoc. Comput. Mach., 22, 2, pp. 261-269, April 1975.

41. Gelenbe, E., and Pujolle, G., "The behavior of a single queue in a general queueing network", Atca Informatica, Vol. 7, pp. 123-136, 1976.

42. Feller, W., An Introduction to Probability Theory and its Applications, Vol. I, 2nd Edition, Wiley, New York.

43. Kemeny, J. G., and Snell, J. L., Finite Markov Chains, Van Nostrand, Princeton, NJ, 1967.

List of Tables

Module Number	Estimated IRM parameters	Estimated First-order Markov transition matrix	
0	0.5063	0.4090	0.5910
1	0.4938	0.6060	0.3940

Table 1. Estimated IRM and FMM parameters for $M = 2$ (averaged over all 4 trace tapes)

Module number	Estimated IRM parameters	Estimated First-order Markov Transition matrix			
0	0.2500	0.2762	0.4917	0.1258	0.1066
1	0.2466	0.1008	0.3094	0.4959	0.0935
2	0.2581	0.1178	0.0974	0.2875	0.4973
3	0.2453	0.5121	0.0909	0.1230	0.2741

Table 2. Estimated IRM and FMM parameters for $M = 4$ (averaged over all 4 trace tapes)

Module Number	Estimated IRM parameters	Estimated First-order Markov transition matrix							
0	0.1305	.2158	.3898	.0575	.0584	.0522	.0894	.0908	.0468
1	0.1238	.0478	.2333	.4453	.0422	.0410	.0704	.0893	.0304
2	0.1262	.0621	.0477	.2432	.4337	.0568	.0342	.0600	.0624
3	0.1185	.0497	.0402	.0646	.2253	.4612	.0615	.0428	.0543
4	0.1180	.0662	.0636	.0512	.0579	.2111	.4609	.0458	.0433
5	0.1249	.0572	.1085	.0589	.0387	.0477	.2075	.4114	.0701
6	0.1334	.0811	.0549	.0462	.0460	.0353	.0576	.2267	.4522
7	0.1246	.4566	.0395	.0451	.0517	.0700	.0392	.0868	.2109

Table 3. Estimated IRM and FMM parameters for $M = 4$ (averaged over all 4 trace tapes)

Number of memory modules	Estimated SFMM parameters															
	P_o	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}
2	0.4016	0.5984														
4	0.2968	0.4992	0.1070	0.1070												
8	0.2218	0.4389	0.0568	0.0604	0.0637	0.0662	0.0485	0.0571								
16	0.1934	0.4094	0.0278	0.0225	0.0323	0.0323	0.0237	0.0292	0.0289	0.0338	0.0284	0.0233	0.0309	0.0310	0.0246	0.0270

Table 4. Estimated SFMM parameters for M = 2,4,8,16(averaged over all 4 trace tapes).

M(=N)	1	2	3	4	5	6	7	8
Strecker	0	0	3.10	4.33	5.06	5.55	5.89	6.15
Exponential	0	-11.11	-12.09	-12.79	-13.18	-13.44	-13.62	-13.75
Asymptotic	0	-4.10	-2.33	-1.73	-1.37	-1.14	-0.97	-0.85
DGF	0	0	0.24	0.15	0.12	0.10	0.09	0.08

Table 5. Percentage errors of various analytical solutions for the RIRM case with $M = N$.

M(=N)	1	2	3	4	5	6	7	8
Simulation results (Sethi and Deo)	1	1.374	1.844	2.347	2.844	3.336	3.843	4.317
DGF	1	1.3714	1.8438	2.3381	2.8393	3.3437	3.8498	4.3569
Percentage error of DGF	0	-0.19	-0.01	-0.38	-0.16	0.23	0.18	0.92

Table 6. Accuracy of the DGF solution for the LRM case

M	Measured Bandwidth (Simulation)	Bandwidth from DGF analysis	Bandwidth from Asymptotic analysis
2	1.8255	1.7888 (-2)	1.7601 (-3.58)
4	2.7217	2.6731 (-1.79)	2.6164 (-3.87)
8	3.2911	3.2921 (0.03)	3.2622 (-0.89)
16	3.6261	3.6359 (0.27)	3.6261 (0.0)

Table 7. Accuracy of the DGF and asymptotic analyses relative to trace-driven simulations ($N = 4$). Percentage errors are indicated in parentheses

M	Measured Bandwidth (Simulation)	Bandwidth from DGF analysis
2	1.7964	1.7888 (-0.42)
4	2.6552	2.6731 (-0.67)

Table 8. Error inherent in the DGF analysis alone relative to simulations driven by reference strings which conform to the assumed (SFMM) model of program behavior (N=4). Percentage errors are indicated in parentheses

M	4	8	16
α	0.1798	0.1653	0.1650
β	0.3923	0.3824	0.3810

Table 9. α and β as calculated from the SFMM parameters for $M = 4, 8, 16$

M	Bandwidth using the 2-parameter model obtained from the SFMM parameters for:			
	M = 4		M = 16	
2	1.7921	(-1.83)	1.7928	(-1.79)
4	2.6731	(-1.79)	2.6738	(-1.76)
8	3.2900	(-0.03)	3.2905	(-0.02)
16	3.6355	(0.26)	3.6358	(0.27)

Table 10. Accuracy of the 2-parameter model relative to trace-driven simulation results (N=4). Percentage errors are indicated in parentheses